



Apresentando a estrutura do Projeto FullStack a ser desenvolvido na disciplina

Vamos apresentar e detalhar a estrutura do Projeto FullStack a ser desenvolvido como principal entrega na Disciplina "Prática Integradora – Tecnologias Disruptivas". Como o nome da disciplina indica, o projeto principal a ser desenvolvido durante a mesma é um projeto integrado, uma aplicação composta por banco de dados, back e front-end. Tal aplicação será composta por um CRUD (Create|Read|Update|Delete) de Projetos, Tarefas e Recursos.

Normalmente, no processo de análise e desenvolvimento de Software, o primeiro passo consiste no entendimento do problema a ser resolvido/solução a ser desenvolvida. Para isso, existem inúmeras técnicas, como análise e refinamento de requisitos, criação de protótipos, etc. No escopo dessa disciplina, iniciaremos esse processo um pouco mais a frente, a partir do Diagrama de Entidades e Relacionamentos, em que temos o Modelo Conceitual das tabelas, seus atributos e relacionamentos, que comporão o banco de dados da aplicação. O DER pode ser visto abaixo, na figura 1.



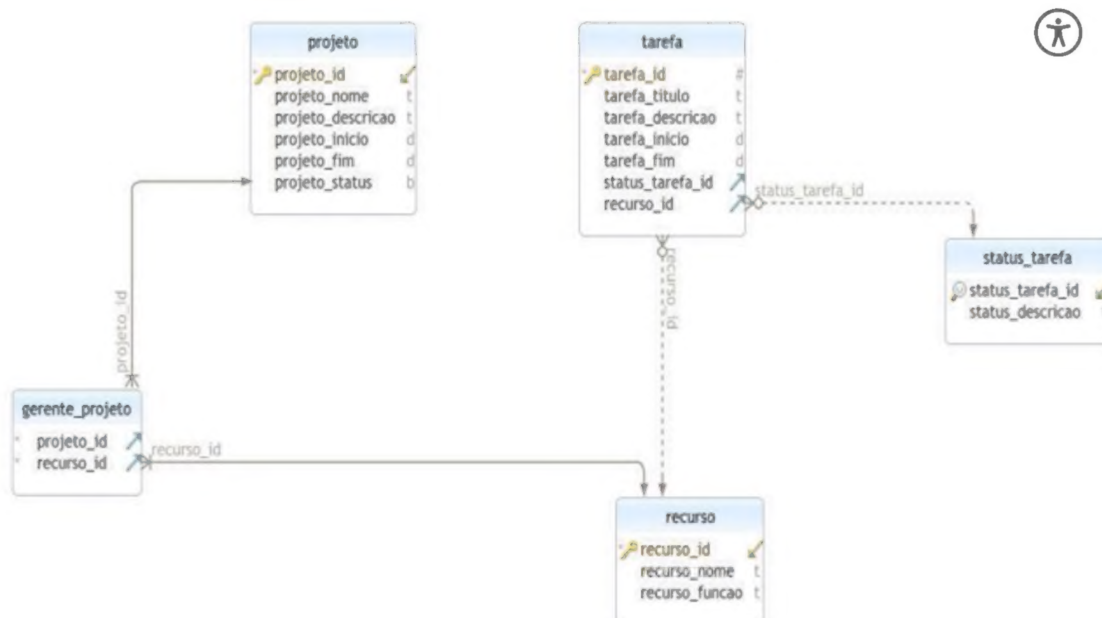


Figura 1: DER do Projeto Integrado

A partir do DER, e ainda em termos de documentação, poderíamos elaborar também o Diagrama de Classes. Entretanto, e considerando que o backend será implementado através de uma API Restful – onde através do Mapeamento Objeto-Relacional (ORM), criaremos as Classes de Entidade que representarão a estrutura vista no DER, e tendo em mente que esse processo de Mapeamento segue algumas regras específicas do Framework a ser utilizado no backend, poderemos omitir essa etapa.

Nesse ponto, e já entrando em alguns detalhes técnicos de implementação, vejamos as tecnologias que serão empregadas em nosso Projeto:

- Banco de Dados: H2 (um banco de dados embarcado). Sobre tal componente da aplicação, você poderá utilizar outro SGBD de sua

escolha, bastando, para isso, alterar as configurações de conexão no backend;



- Backend: Nosso projeto será, como já mencionado, em termos de back, uma API Restful. Para confecção da API utilizaremos o framework Spring Boot. Logo, utilizaremos a linguagem Java;

- Frontend: Em relação ao front, nossa aplicação será uma aplicação web, onde usaremos o framework React.js – um framework Javascript.

Além das tecnologias acima, utilizaremos ainda outras, relacionadas ao deploy do projeto, como Git Lab e Docker, por exemplo.

A arquitetura do nosso projeto pode ser vista abaixo, na figura 2.

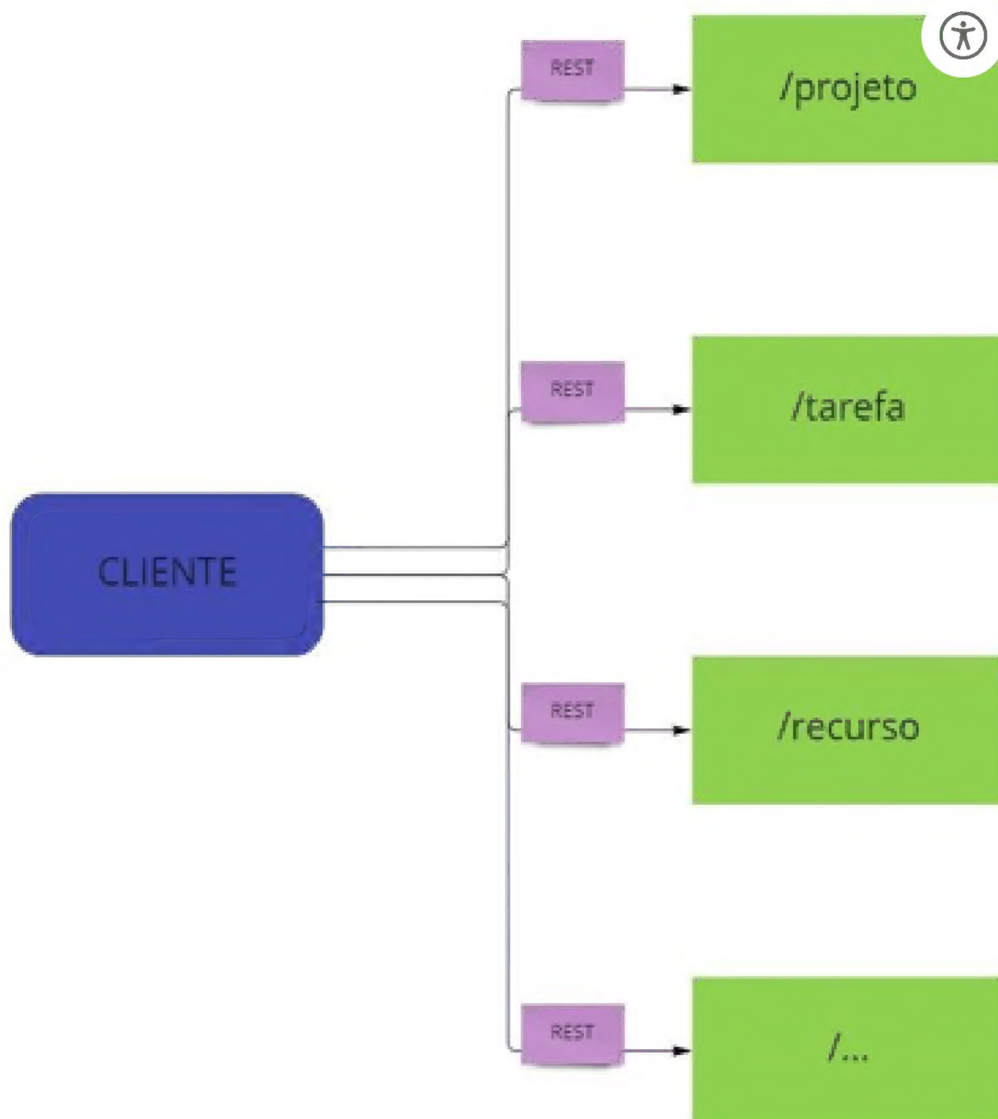


Figura 2: Arquitetura da Aplicação

Como demonstrado acima, a arquitetura a ser utilizada é bastante simples: o cliente, em nosso caso o front-end da aplicação, acessará os endpoints disponibilizados pela API. A partir desse acesso, todas as operações CRUD poderão ser realizadas. Embora funcional e ainda bastante utilizada, há modelos de arquitetura mais modernos e muito utilizados atualmente, como o Gateway de API e o BFF (Backend for Frontend). Mesmo não utilizando tais arquiteturas em

nosso projeto, veremos um pouco sobre as mesmas a partir de agora.

O Gateway de API (ou API Gateway) é um padrão de projeto que estende o modelo normalmente utilizado em APIs (figura 2). Nesse modelo, uma nova camada é inserida entre o cliente e a API, chamada de Gateway. Com isso, essa camada passa a ser um ponto único de acesso à API. Outro conceito nesse padrão é a utilização de Microsserviços (que, de forma rasa, podem ser considerados uma especialização das APIs, com maior separação e individualização por responsabilidade dos serviços executados). A figura 3 demonstra como ficaria a nossa arquitetura utilizando a API Gateway.

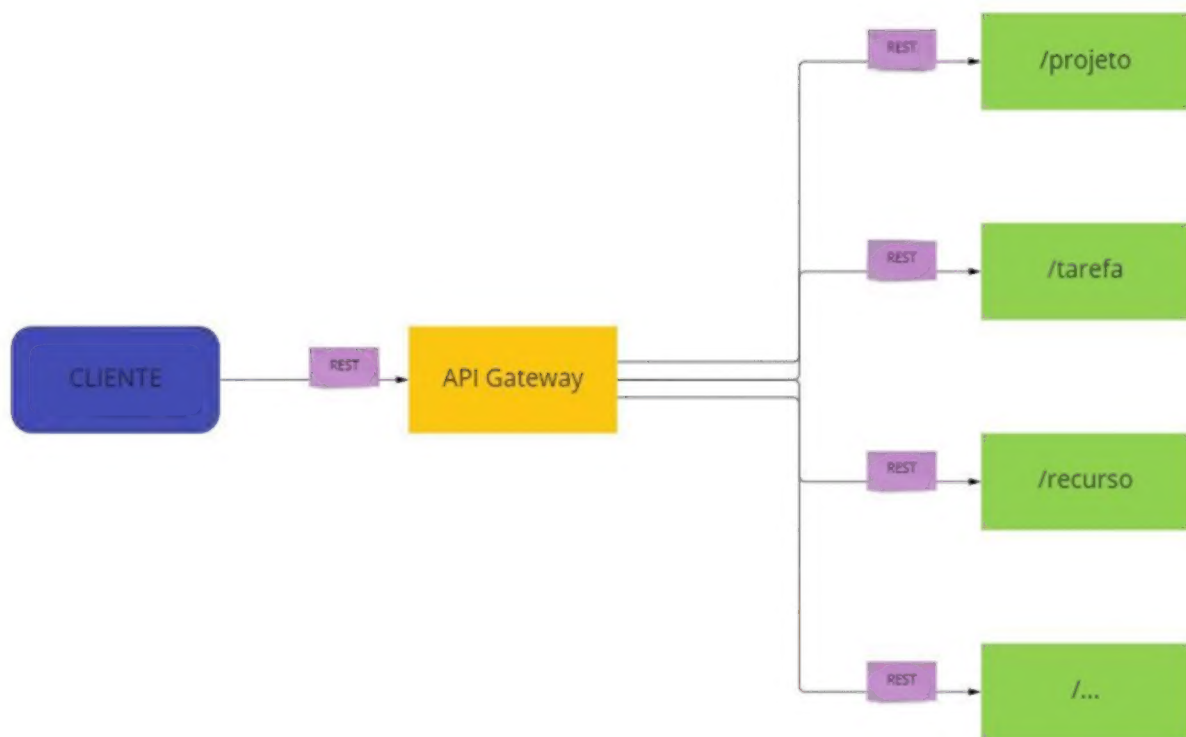


Figura 3: Arquitetura API Gateway

Além dos detalhes já mencionados, cabe ainda destacar que, nessa arquitetura, a API Gateway, além de um ponto único de acesso aos

microsserviços que compõem a aplicação, é também a única interface pela qual os clientes terão acesso aos serviços. Ess .ima característica, inclusive, é o que difere tal arquitetura da próxima que veremos, a BFF (Backend From Frontend). Nesse padrão de projeto temos a mesma topologia da API Gateway, exceto pela existência de mais de uma interface para acesso por parte dos clientes. Aqui, a camada intermediária será diferente de acordo com os diferentes tipos de cliente que a acessarem. Podemos ter, por exemplo, um BFF para Clientes Web. Um outro para Clientes Mobile e assim por diante. Cabe destacar também que cada um desses BFFs não precisam ser iguais. Cada um deles pode conter regras próprias, específicas. A figura 4, abaixo, mostra essa arquitetura, caso aplicada em nosso projeto.

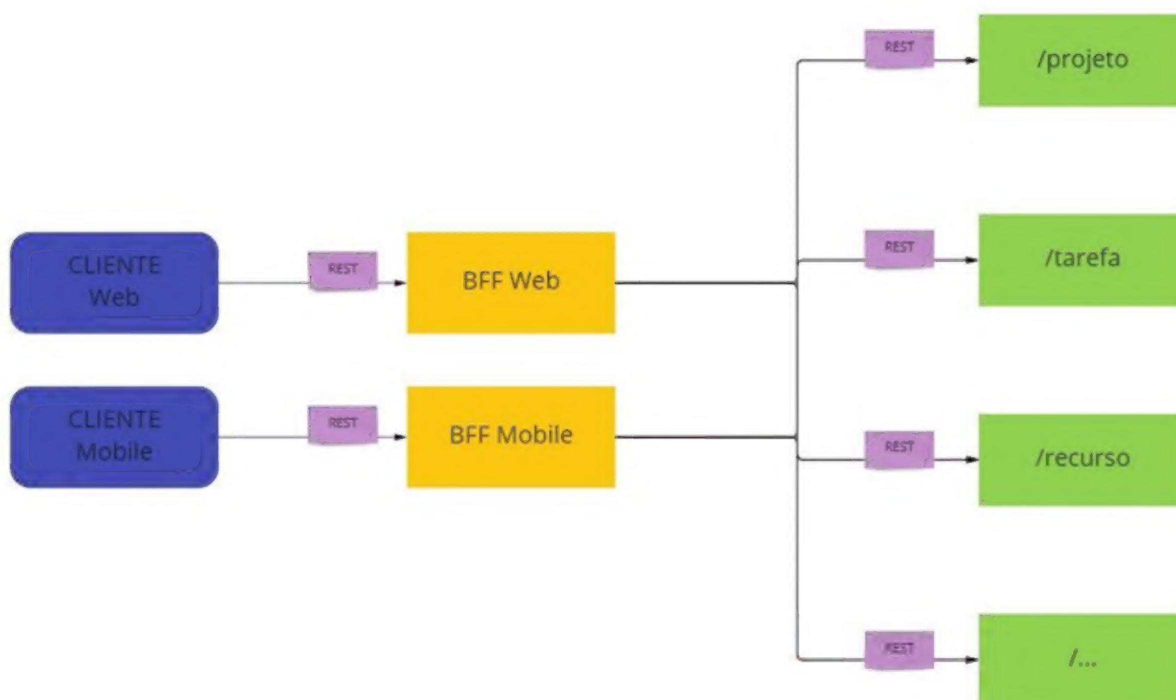


Figura 4: Arquitetura BFF

Após a apresentação da arquitetura de nosso projeto, além da breve conceitualização de outros modelos de arquitetura existentes, chegamos ao fim deste módulo. Outros detalhes, como bibliotecas adicionais, estrutura de pacotes e pastas, entre outros, serão vistos mais adiante.

Atividade Extra

Um tema abordado ao longo desse módulo, mas não aprofundado, foram os microsserviços. Para saber mais sobre tal assunto, assista ao vídeo abaixo:

Link: https://www.youtube.com/watch?v=j3XufmvEMiM **(Acesso em 28/09/2022)**

Referência Bibliográfica

HOQUE, Shama. **Full-Stack React Projects.** Birmingham, UK: Packt Publishing, 2020.

Atividade Prática Módulo 7

Título da Prática: Criando a estrutura do Projeto Integrado FullStack.

Objetivos: Criar a estrutura de pastas dos projetos de back e frontend, além da instalação das ferramentas e bibliotecas, necessários para codificação do Projeto Integrado.

Materiais, Métodos e Ferramentas: IDE (Spring Tool Suite; Jetbrains Intellij; etc)

Atividade Prática

Ao longo das atividades práticas resolvidas até aqui, tivemos a oportunidade de criarmos uma (ou até mesmo mais de uma) API Restful com o Spring Boot. Nesse sentido, você já criou uma estrutura de pastas (ou packages) de acordo com a organização em camadas utilizada pelo Spring e também configurou recursos extras como Segurança e Documentação. Ao longo dessa atividade, vamos construir, então, a estrutura básica de nosso projeto de frontend, no qual utilizaremos o React Js. Além disso, vamos conferir se temos, em nosso computador, tudo o que precisamos para codificação do Projeto Integrado FullStack. Para concluir essa atividade, você precisará cumprir cada etapa a seguir:

1. Ter uma IDE instalada em seu computador para codificação do backend. Recomendo uma dessas duas: Spring Tool Suite ou IntelliJ (ambas gratuitas);

2. Ter uma IDE instalada em seu computador para codificação do frontend. Recomendo o VS Code (gratuito);

3. Ter uma ferramenta instalada em seu computador para testar a API. Recomendo o Insomnia (gratuito). Nesse sentido, você poderá usar, como alternativa, o Swagger UI (que, além de documentar, permite também testar a API);

4. Ter o programa de gerenciamento de pacotes, NPM, necessário para criar projetos React;

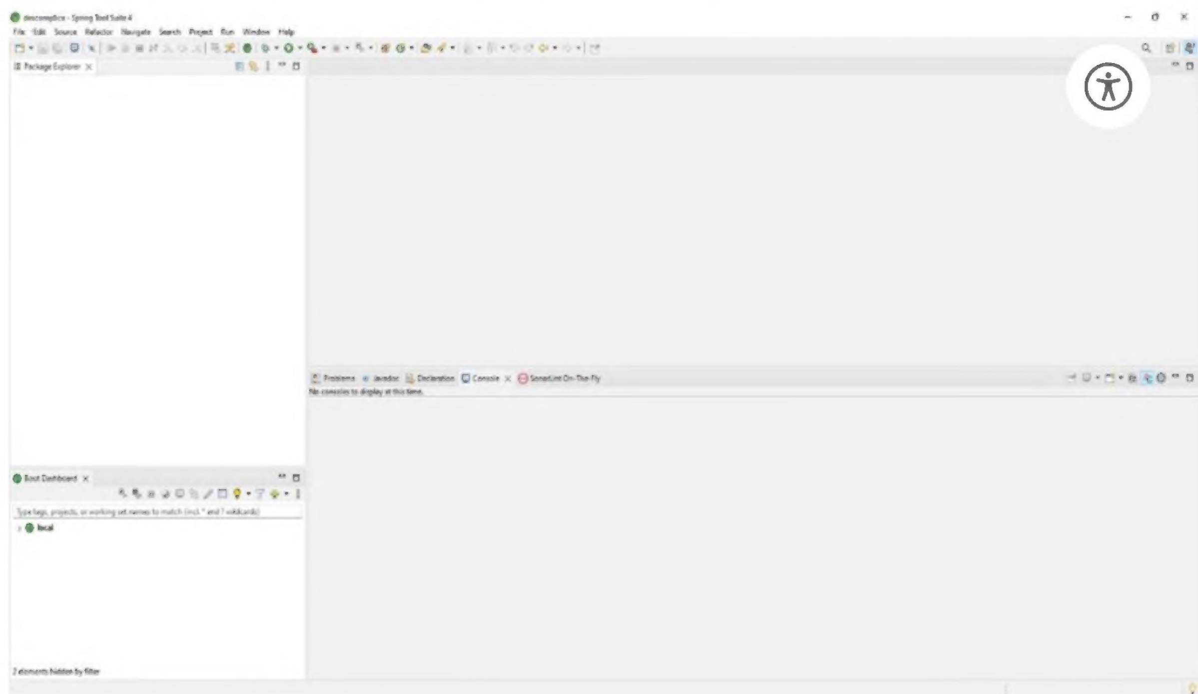
5. Criar a estrutura do projeto de backend, tendo como ponto de partida o start.spring.io . Após isso, lembre-se de já criar as camadas da API;

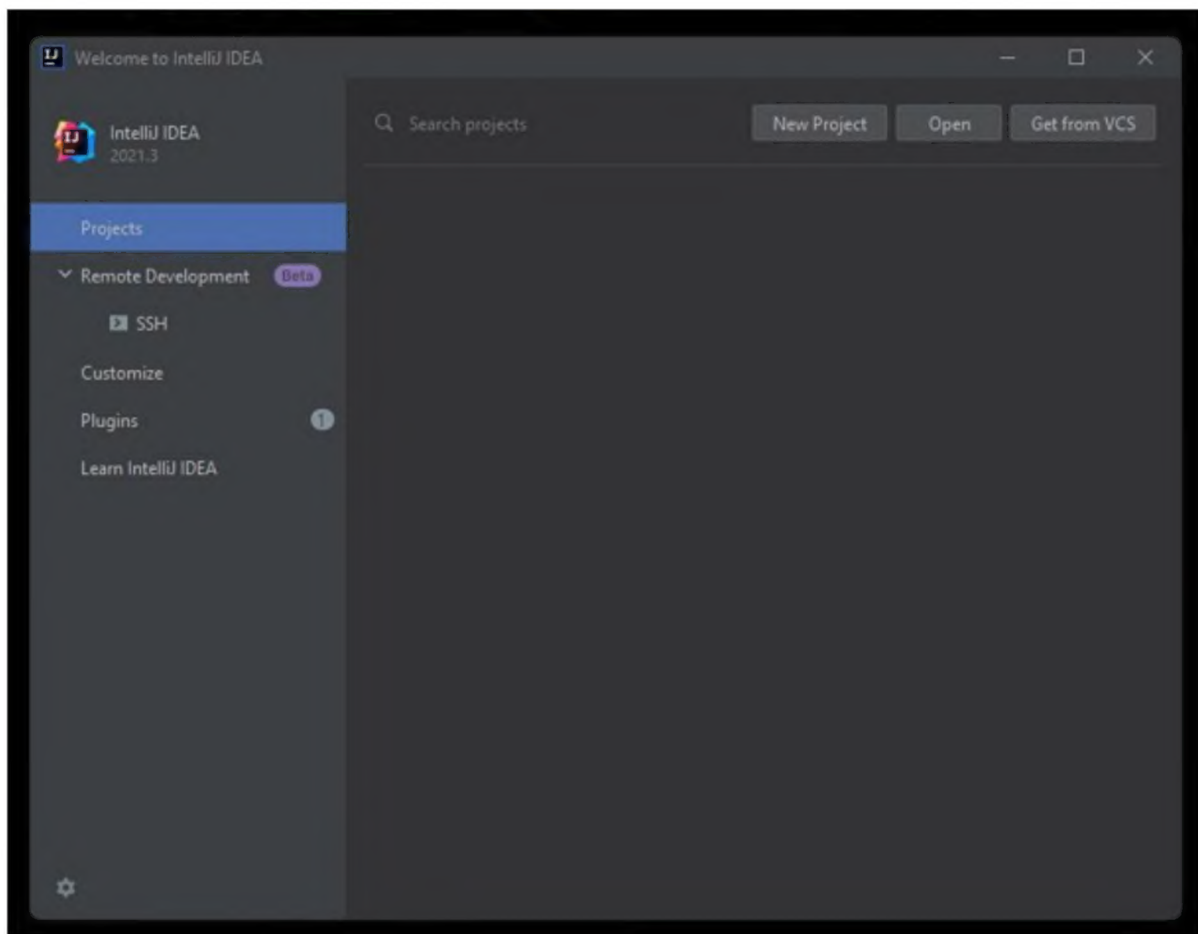
6. Criar um primeiro projeto React e testá-lo no navegador.

Gabarito Atividade Prática

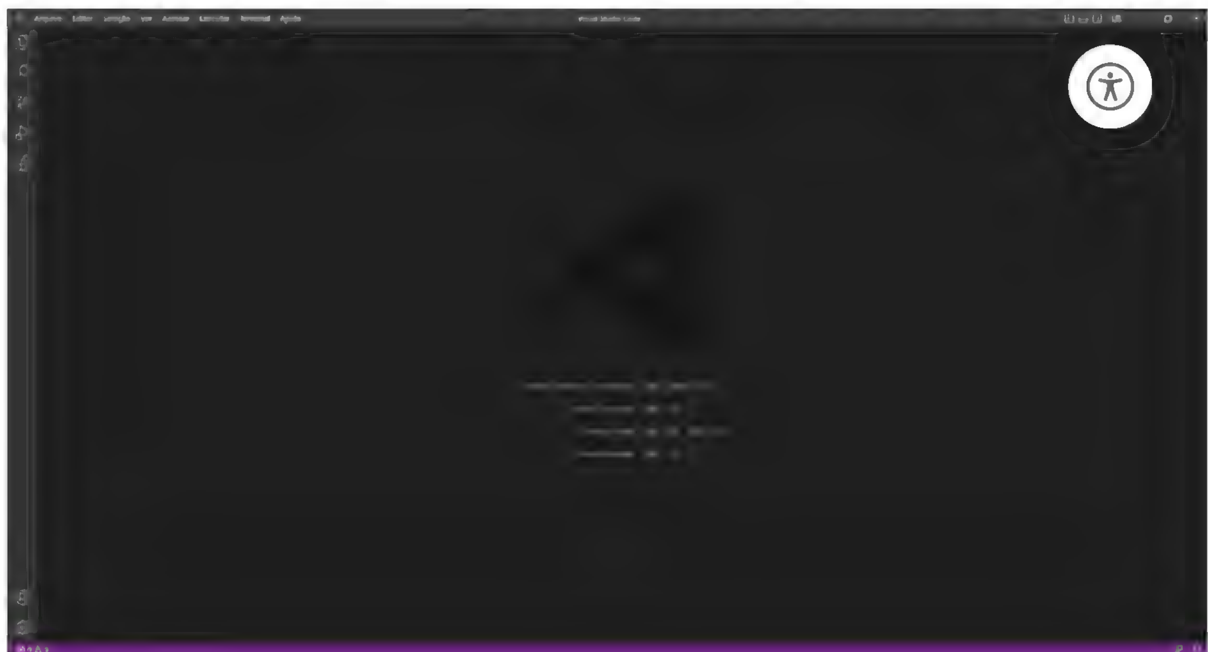
Segue o gabarito, com algumas evidências que poderão ser utilizadas para comprovar a execução com êxito das atividades, conforme passo a passo acima:

1. Prints das telas iniciais do Spring Tool Suite e do IntelliJ. Essas evidências comprovam que uma dessas IDEs foi instalada no computador do aluno e se encontra funcional, pronta para uso.





2. Print da tela inicial do VS Code. Tal evidência comprova que essa IDE foi instalada no computador do aluno e se encontra funcional, pronta para uso.



3. Print da tela inicial do Insomnia. Tal evidência comprova a instalação da ferramenta pelo aluno em seu computador.



4. Há várias formas de se instalar o gerenciador de pacotes NPM. Tal instalação, inclusive, varia de acordo com o Sistema Operacional. Entretanto, independente da forma de instalação ou do SO, para comprovar que o NPM foi instalado com sucesso basta rodar o comando abaixo, via linha de comando:



npm -v

Como saída, esse comando apresenta a versão instalada do npm, conforme print abaixo:

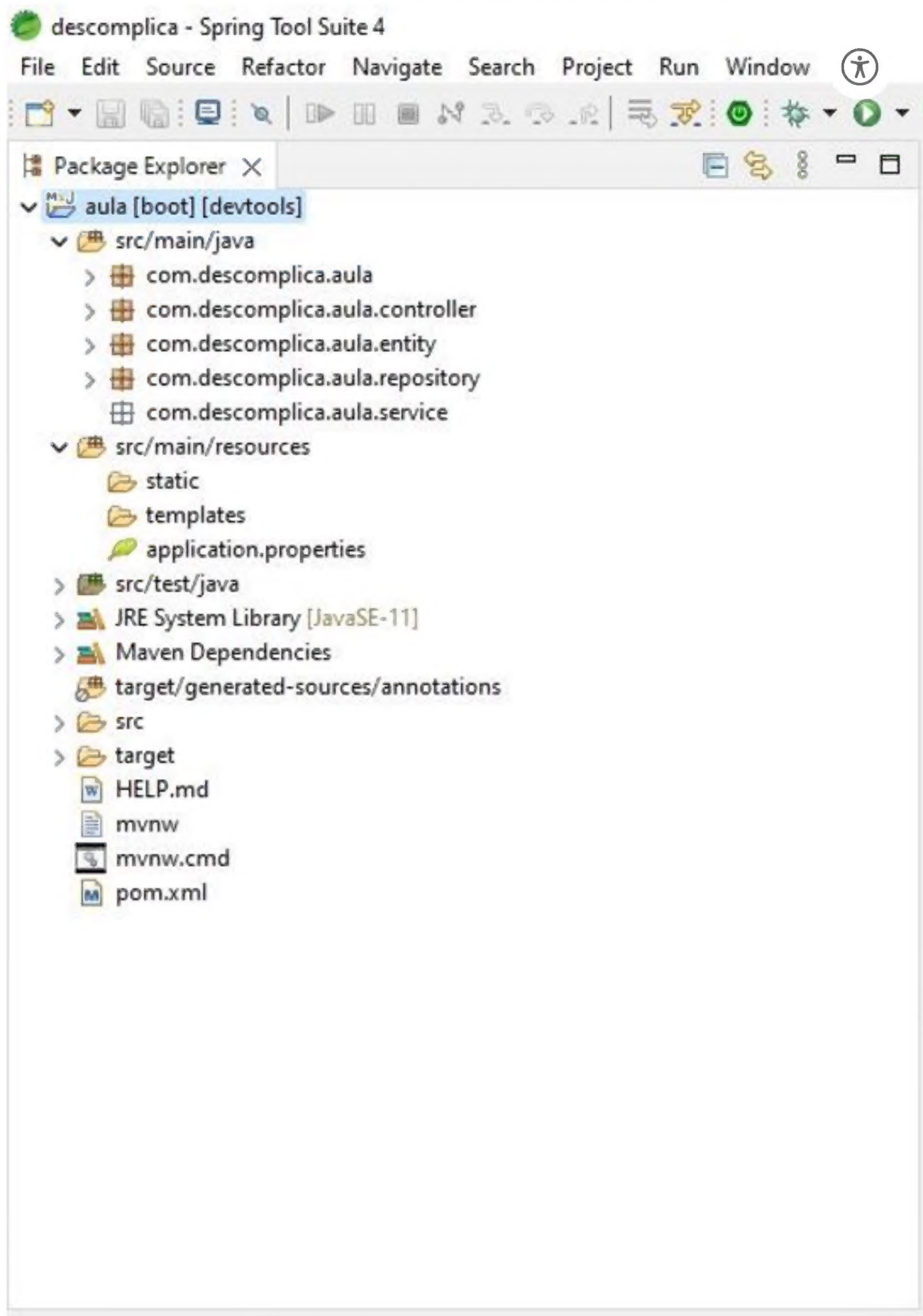
```
C:\WINDOWS\system32\cmd.
Microsoft Windows [versão 10.0.22000.856]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\aopai>npm -v
8.4.0

C:\Users\aopai>
```

5. A partir do site start.spring.io é possível configurar e baixar a estrutura inicial de um projeto que faz uso do Spring Boot. A imagem a seguir, mostrará a partir de um print, de tal site, o exemplo de algumas configurações e bibliotecas selecionadas:

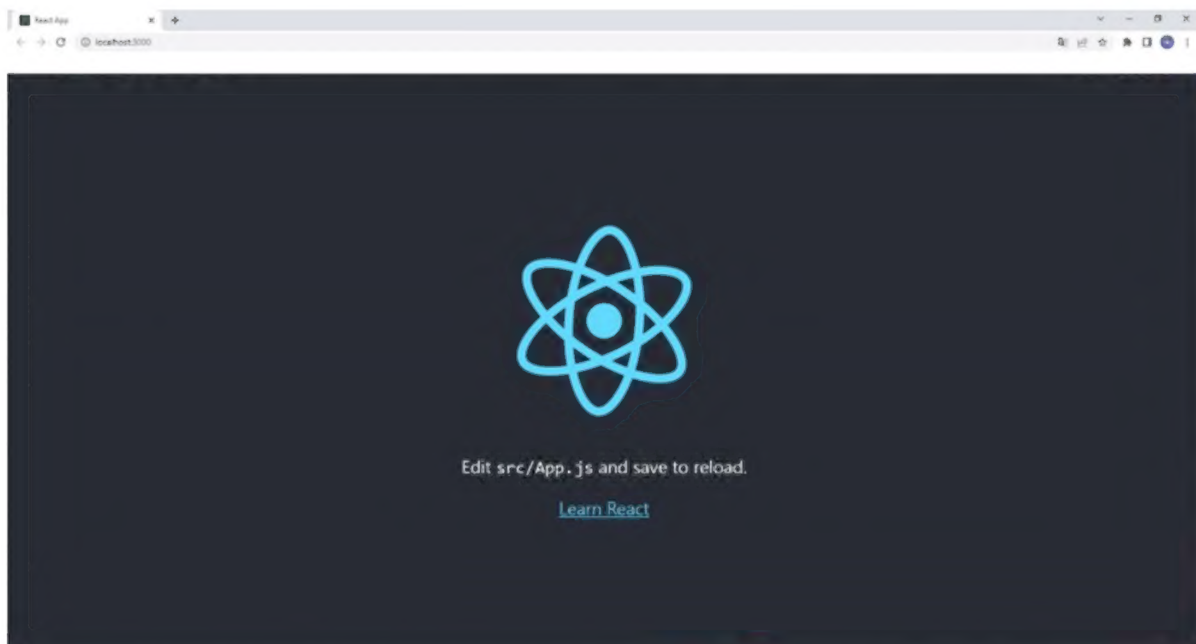
O Spring sugere uma organização do projeto da API em diferentes camadas. O print abaixo mostra tal estrutura, vista a partir da configuração do projeto (baixado a partir do start.spring.io, conforme print anterior) na IDE Spring Tool Suite.



6. O processo de criação de um projeto React, por padrão, consiste na execução do comando abaixo, via linha de comando:

npx create-react-app nome-do-projeto

Ao término da execução desse comando, uma nova pasta será criada dentro do sistema de arquivos, no computador do al. Tal pasta terá o mesmo nome dado ao projeto (no exemplo do comando, será “nome-do-projeto”). Após a conclusão do comando em questão, deverá ser acessada a pasta do projeto criado, também via linha de comando. Estando na pasta do projeto, será possível executar um outro comando para iniciar a aplicação, o “npm start” . Esse segundo comando realizará algumas tarefas e, ao final, abrirá uma janela do navegador (default) mostrando a tela a seguir:



Ir para exercício